陰関数描画に関する一つの試み(数式処理における理論とその応用の研究)

AUTHOR(S):

近藤, 祐史; 三好, 善彦; 齋藤, 友克

数理解析研究所講究録
920巻 1995年 165-172

18.

陰関数描画に関する一つの試み

近藤祐史 (詫間電波高専)
三好善彦 (埼玉女子短大)
齋藤友克 (上智大学)

18.1 はじめに

計算機を用いて様々な問題を解決しようとする場合計算の結果を可視化することは、現象の理解、背景の洞察もしくは新しい課題の発見等のために欠くべからざるプロセスになっている。特に数式で与えられた結果の幾何的イメージを把握出来れば、解の存在範囲や交点の個数配置などが容易に見る事が出来、最終的な答えを導き出すことに役にたつ場合がある。

関数描画としては、基本的な出発点として大別して追跡法と全画面検索法がある。我々の基本となる方針は全画面検索法を前提として考察する。本稿では関数の描画に対する基本的な出発点を提起したい。

18.2 関数の描画とは

関数の描画とは、与えられた領域 D での関数 $f=0$ の解を必要とされる出力装置 (ディスプレイ、プリンター等) に出力することとする。

与えられた関数を表現する場合、関数の計算精度、描画精度は描画する装置の解像度に依存する。そのため描画とは、関数と定義域および解像度と呼ばれるものにによって定められる描画関数を計算することである。以下描画に対する基本的な定義として解像度ならびに描画関数 χ の定義を述べる。

18.2.1 定義

以下ここで扱う関数 f は、R^n の連結部分集合 D 上で定義された関数であって D 上連続な関数とする。

■解像度

画像の解像度とは定義域 D の部分集合の族 $S = \{S_i, | S_i$連結 $i = 1, \cdots, m\}$ であって次のものをいう。

定義 18.2.1. 集合 S が D 上定義された関数 f の解像度とは

$$D = \bigcup_{i=1}^{n} S_i, \quad S_i \bigcap S_j = \phi \quad i \neq j$$

■描画関数

描画関数として領域、解像度、非描画関数によって定められる様々な特性関数、character $\chi(D, S)$ と呼ばれるものを定義する。

定義 18.2.2. D 上の解像度 S による character $\chi(D, S)(f)$ とは、

1. $\chi : S \rightarrow \{0, 1\}$
2. $\chi(S_i) = 0$ ならば S_i の任意の元 x に対し $f(x) \neq 0$

定義 18.2.3. D 上の S における faithful character χ とは

1. χ は 関数 f の S における character
2. $\chi(S_i) = 1$ ならば S_i のある元 x が存在し $f(x) = 0$ である

関数の描画としては、この faithful character を計算できれば問題は、解決している。しかし、関数が代数関数に限っても現在のところ万能な計算アルゴリズムは存在しない。そこで、十分実用に耐える character として条件をを弱めた次のような boundary character を提唱したい。

定義 18.2.4. D 上の S による boundary character $\chi(D, S)(f)$ とは

1. $\chi : S \rightarrow \{0, 1\}$
2. $\chi(S_i) = 0$ ならば $\overline{S_i}$ の任意の元 x に対し $f(x) \neq 0$ である。

ここで $\overline{S_i}$ は集合 S_i の境界からなる集合である。

18.3 ifplot

18.3.1 First Step

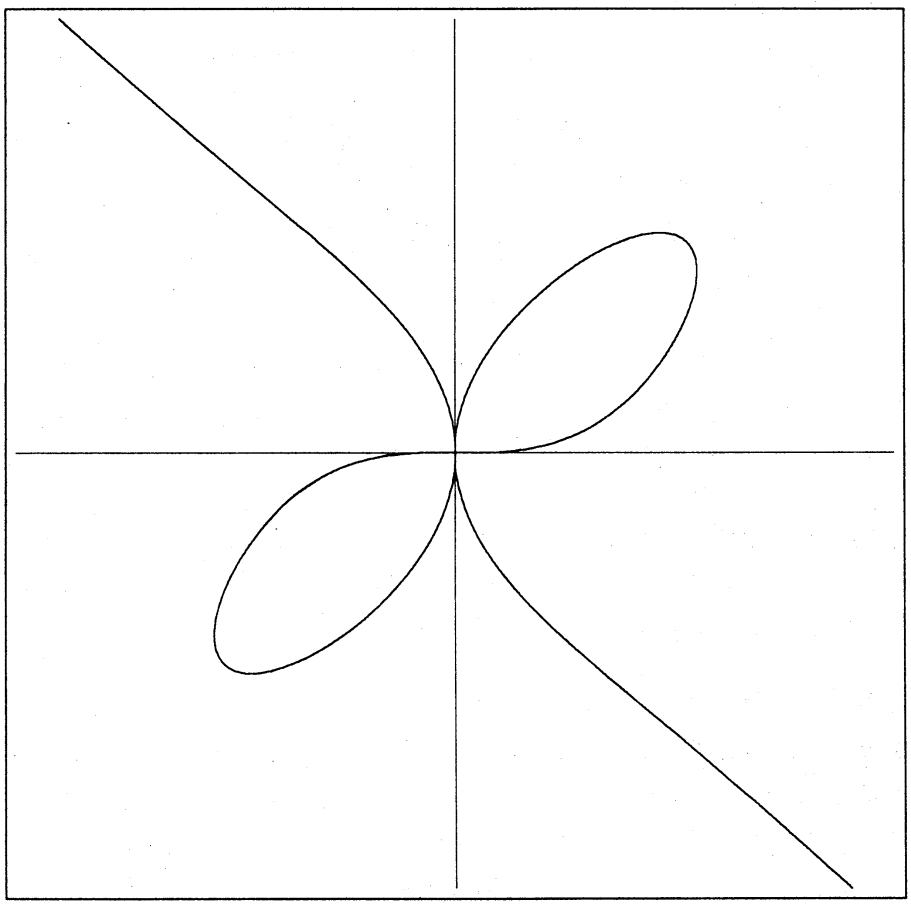
ifplot は、平面領域を小さな矩形に分割しその一つ一つを解像度 S_i と捉えることにより、描画関数 boundary character を求め関数描画を行っている。ifplot の基本的な考え方は次の**中間値の定理**に基づいている。

定理 18.3.1. 中間値の定理 ある区間において連続な関数 $f(x)$ が、この区間に属する点 a, b において相異なる値 $f(a) = \alpha$, $f(b) = \beta$ をとるとき、α, β の中間にある任意の値を μ とすれば、$f(x)$ は a, b の中間のある点 c において、この値 μ をとる。

解像度に対応する矩形の境界点 (x_0, y_0), (x_0, y_1), (x_1, y_0), (x_1, y_1) の関数値 $f(x, y)$ の符号を判定することによって解像度 S_i の boundary character を求めている。

例えば、$f(x_0, y_0)$ と $f(x_1, y_0)$ の符号が異なれば ($f(x_0, y_0) < 0$ と $f(x_1, y_0) > 0$ など)、中間値の定理により x_0 と x_1 の区間内で必ず $f(x, y_0) = 0$ となる。

ifplot の出力例 1



$$x^5 - 2x^2y + y^5$$

18.3.2 Second Step

boundary character を求める時に中間値の定理を用いているが、その時の区間内に奇数個の解が関数に存在すれば良いのであるが (奇数個の解を持てば必ず区間の端の点の関数の符号は異なる)、偶数個の解が関数に存在する時が問題となる。すなわち、偶数個の解を持てば区間の端の点の関数値の符号は同じとなってしまい、中間値の定理を利用したこの方法では解を持たないと判断してしまうからである。

実際問題として、解像度に依存する S_i は出力装置の最小の単位であるため、視覚上はそれほど問題にならないと言えるが、数学的に考えた場合は問題となるであろう。

■ Sturm の定理の利用

中間値の定理のみでは解決できない場合の問題点を解決するための方法として、boundary character を求める時に、Sturm 列を用いれば正確に区間内の解の個数を判定することが可能である。

定義 18.3.2. Sturm 列 $f(x)$ を実係数方程式とする。ただし、$f(x)=0$ は重根を持たないとする。従って、$f(x)$ とその導関数 $f'(x)$ は共通因子を持たない。

$f(x)$ と $f'(x)$ にユークリッドの互除法を行ってその最大公約数を求める。

$$\begin{aligned} f(x) &= q_1(x)f'(x) - f_2(x) \\ f'(x) &= q_2(x)f_2(x) - f_3(x) \\ f_2(x) &= q_3(x)f_3(x) - f_4(x) \\ &\cdots \\ f_{m-2}(x) &= q_{m-1}(x)f_{m-1}(x) - f_m \end{aligned}$$

ここで、f_m は 0 でない定数を表し、このようにして得られた整式の列

$$f_0(x), f_1(x), f_2(x), \cdots, f_m (\text{ただし、} f_0(x) = f(x), f_1(x) = f'(x)$$

のことを多項式の Sturm 列と定義する。

定理 18.3.3. Sturm の定理 $a < b$ で a, b は $f(x)$ の根でないならば、a と b の間にある $f(x)$ の実根の個数は $V(a) - V(b)$ に等しい。ただし、$V(a)$ は Sturm 列

$$f_0(a), f_1(a), f_2(a), \cdots, f_m$$

の符号の変化の数とする。

中間値の定理の場合は、ある区間の端の点の関数値の符号が異なればその区間内に必ず存在することが言えるが、この定理の場合は、Sturm 列を求めることにより “その区間内の解の個数までも判定することが出来るという” ことである。

18.4 領域内の零点判定

領域 D において零点が存在するか否やの判定を行うアルゴリズムは、一般的には極めて困難な問題である。本アルゴリズムは、領域内部に零点が存在する必要条件を求める。とくに、孤立特異点を表現する方法は現在一般的なアルゴリズムは提案されていない。

18.5 区間演算アルゴリズム描画

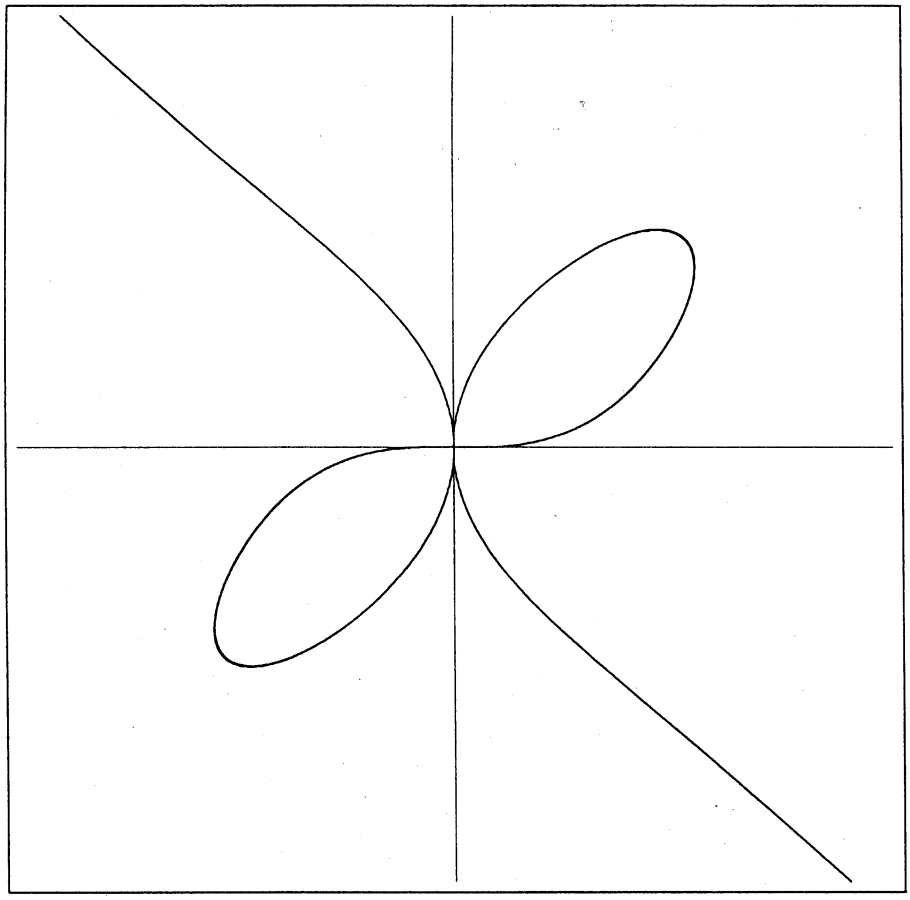
領域 D を $D = \{(x,y)|a_0 < x < a_n, b_0 < y < b_m\}$ によって与えられているとする。解像度を定義する $S_{i,j}$ は、区間 (a_0, a_n) を n 個に分割した、$a_0, a_1, \cdots, a_n$ 列と区間 (b_0, b_m) を m 個に分割した、$b_0, b_1, \cdots, b_m$ により $S_{i,j} = \{(x,y)|a_{i-1} < x < a_i,\ b_{j-1} < y, b_j\}$ と与えられているとする。このとき X_i, Y_j を各々区間数 $X_i = (a_{i-1}, a_i)$, $Y_j = (b_{j-1}, b_j)$ とすれば、つぎの定理が成り立つ。

定理 18.5.1. 領域 $\{(x,y)|\ a_{i-1} \leq x \leq a_i,\ b_{j-1} \leq y \leq b_j\}$ 内において関数 $f(x,y)$ が零点を持てば、$f(X_i, Y_j)$ はゼロを含む区間数である。

このことを利用する事により比較的容易に領域内の零点の存在を推定することが出来る。特に孤立特異点を推定するアルゴリズムとしては現在このアルゴリズムのみである。しかしこのアルゴリズムは、計算量が増大するてんが問題ではある。

18.5.1 区間演算利用の例

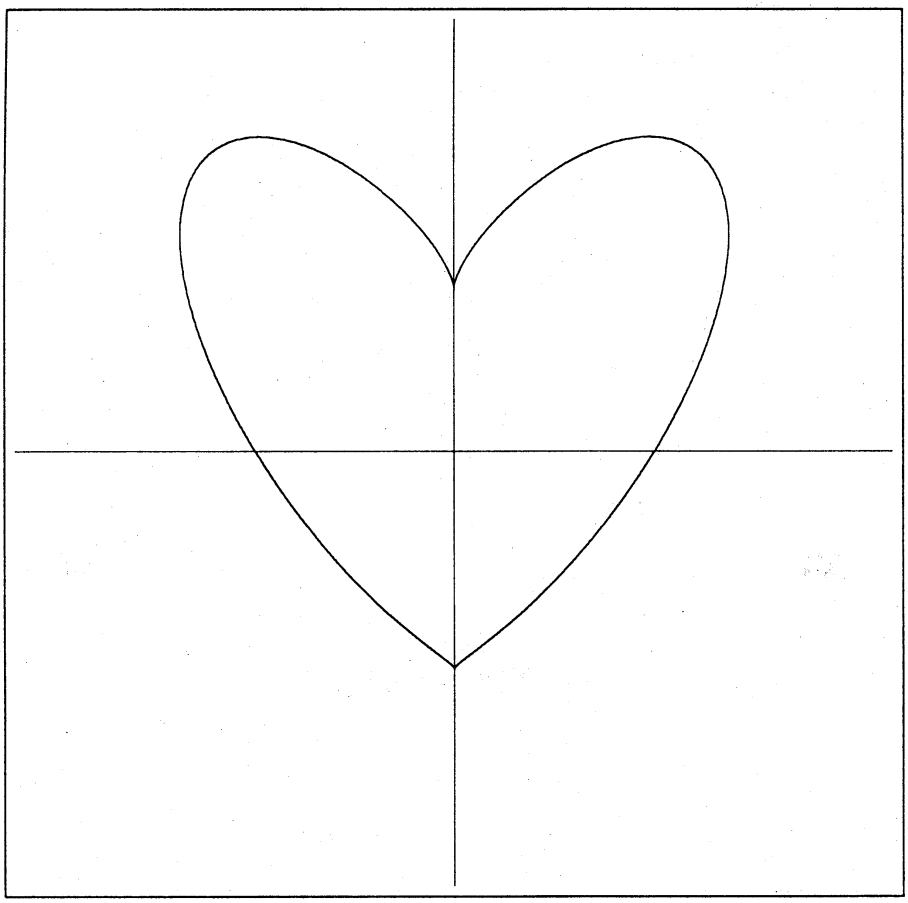
例 1 で計算した図を区間演算を利用した方法で表示すると



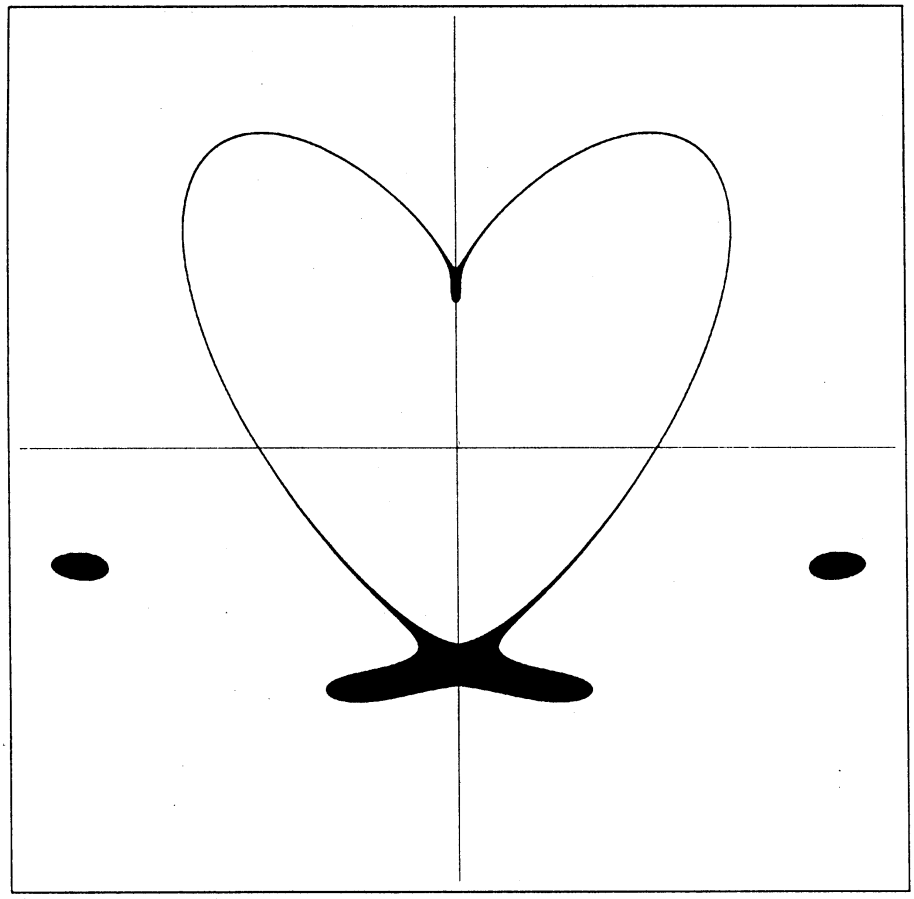
この場合は、孤立特異点が無いため通常の ifplot と同程度のグラフが求まる。しかし、孤立特異点が存在する場合は、次の例の様になる。表示する関数は heart 型関数の式

$$93392896/15625 * x^6 + (94359552/625 * y^2 + 91521024/625 * y - 249088/125) * x^4 +$$
$$(1032192/25 * y^4 - 36864 * y^3 - 7732224/25 * y^2 - 207360 * y + 770048/25) * x^2 +$$
$$65536 * y^6 + 49152 * y^5 - 135168 * y^4 - 72704 * y^3 + 101376 * y^2 + 27648 * y - 27648$$

である。



ifplot による出力



区間演算による出力

この図より孤立点が 4 個存在する可能性がある。Gröbner Base を用いて計算すると。

$$\begin{cases} 351 * y + 386 = 0 \\ 123201 * x^2 - 17150 = 0 \end{cases} \qquad \begin{cases} 76 * y + 41 = 0 \\ 2888 * x^2 - 8575 = 0 \end{cases}$$

の各々の解が孤立特異点であることが判明する。竹島卓 (富士通情報研)